On the Analysis of Population Protocols

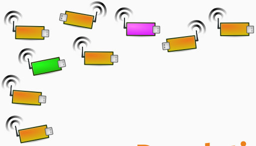
Michael Blondin

Technical University of Munich

Overview

Population protocols: distributed computing model for massive networks of passively mobile finite-state agents

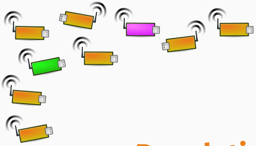
Overview



Population protocols: distributed computing model for massive networks of passively mobile finite-state agents

Can model *e.g.* networks of passively mobile sensors and chemical reaction networks

Overview



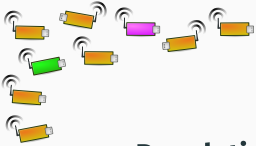
Population protocols: distributed computing model for massive networks of passively mobile finite-state agents

Can model *e.g.* networks of passively mobile sensors and chemical reaction networks

Protocols compute predicates of the form $\varphi\colon \mathbb{N}^d \to \{0, 1\}$

e.g. if φ is unary, then $\varphi(n)$ is computed by n agents

Overview



Population protocols: distributed computing model for massive networks of passively mobile finite-state agents

This talk:

- Automatic verification and testing
- Study of the minimal size of protocols

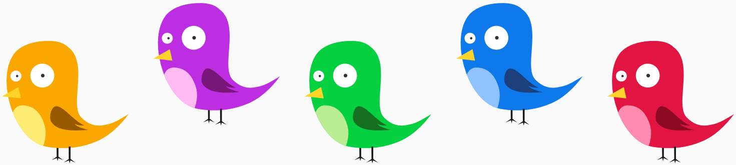
Population protocols

Angluin *et al.* PODC'04

- anonymous mobile agents with very few resources
- agents change states via random pairwise interactions
- each agent has opinion true/false
- computes by stabilizing agents to some opinion

Population protocols

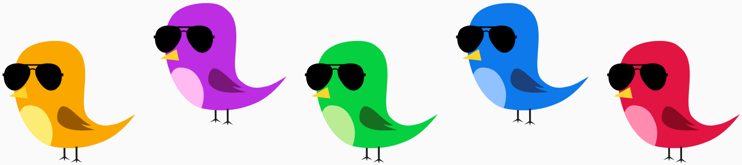
- anonymous mobile agents with very few resources
- agents change states via random pairwise interactions
- each agent has opinion true/false
- computes by stabilizing agents to some opinion



Population protocols

Angluin *et al.* PODC'04

- anonymous mobile agents with very few resources
- agents change states via random pairwise interactions
- each agent has opinion true/false
- computes by stabilizing agents to some opinion

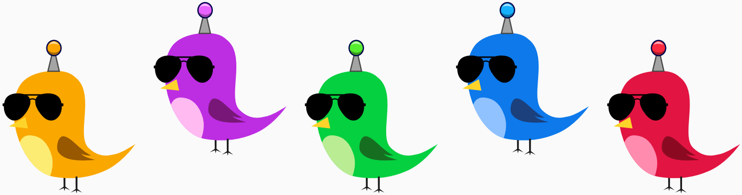


Population protocols

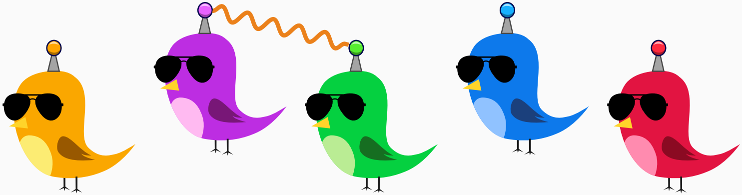
Angluin *et al.* PODC'04

- anonymous mobile agents with very few resources
- agents change states via random pairwise interactions
- each agent has opinion true/false
- computes by stabilizing agents to some opinion

- anonymous mobile agents with very few resources
- agents change states via random pairwise interactions
- each agent has opinion true/false
- computes by stabilizing agents to some opinion



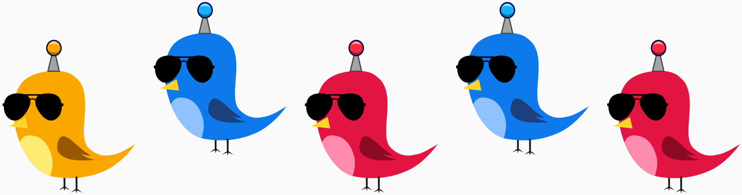
- anonymous mobile agents with very few resources
- agents change states via random pairwise interactions
- each agent has opinion true/false
- computes by stabilizing agents to some opinion



Population protocols

Angluin *et al.* PODC'04

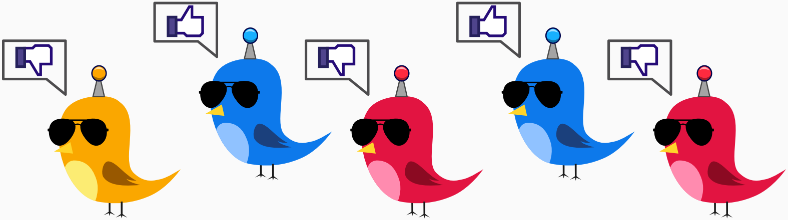
- anonymous mobile agents with very few resources
- agents change states via random pairwise interactions
- each agent has opinion true/false
- computes by stabilizing agents to some opinion



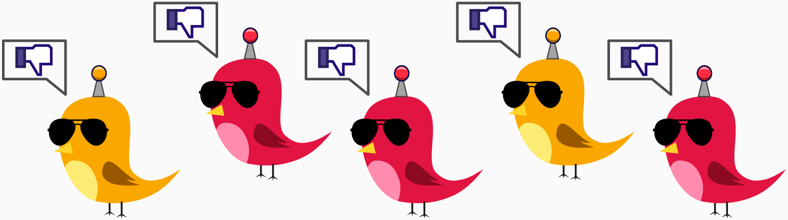
Population protocols

Angluin *et al.* PODC'04

- anonymous mobile agents with very few resources
- agents change states via random pairwise interactions
- each agent has opinion true/false
- computes by stabilizing agents to some opinion

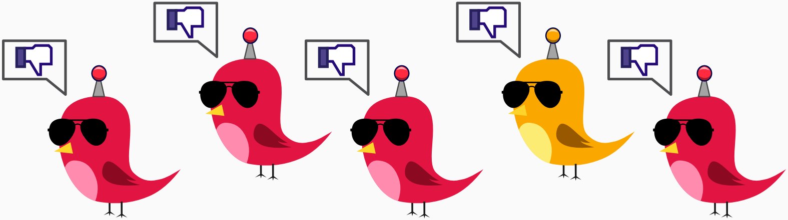


- anonymous mobile agents with very few resources
- agents change states via random pairwise interactions
- each agent has opinion true/false
- computes by stabilizing agents to some opinion



Population protocols

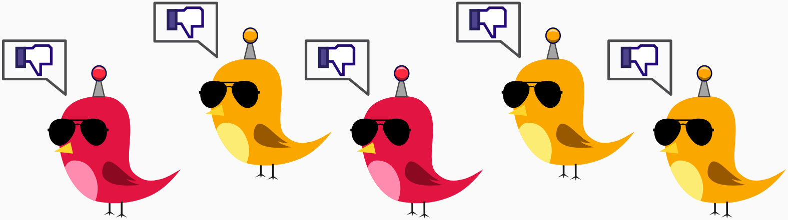
- anonymous mobile agents with very few resources
- agents change states via random pairwise interactions
- each agent has opinion true/false
- computes by stabilizing agents to some opinion



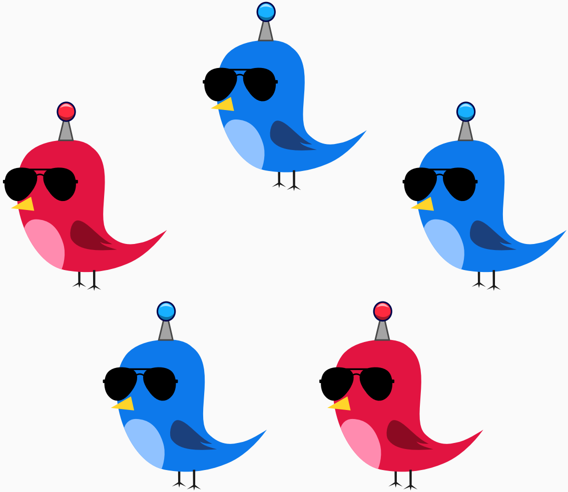
Population protocols

Angluin *et al.* PODC'04

- anonymous mobile agents with very few resources
- agents change states via random pairwise interactions
- each agent has opinion true/false
- computes by stabilizing agents to some opinion



More blue birds than red birds?

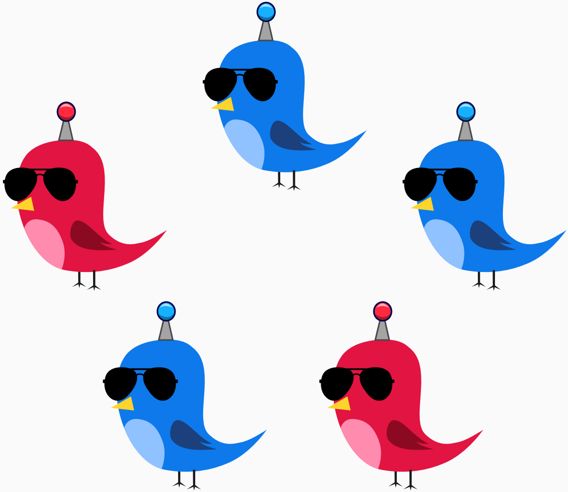


Example: majority protocol

More blue birds than red birds?

Protocol:

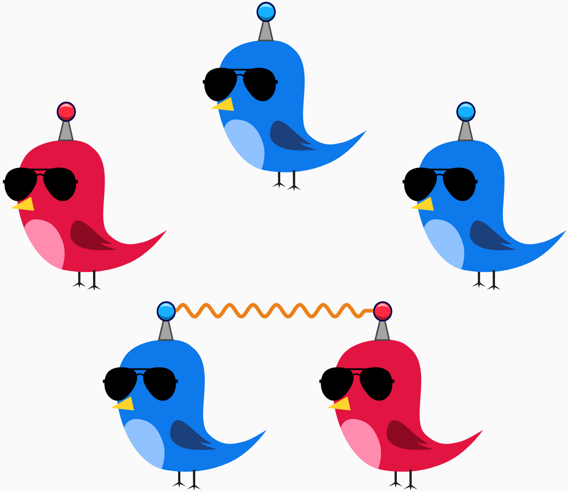
- Two large birds of different colors become small
- Large birds convert small birds to their color



More blue birds than red birds?

Protocol:

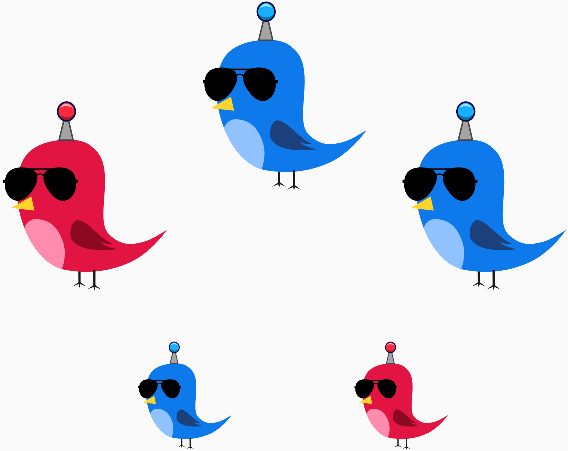
- Two large birds of different colors become small
- Large birds convert small birds to their color



More blue birds than red birds?

Protocol:

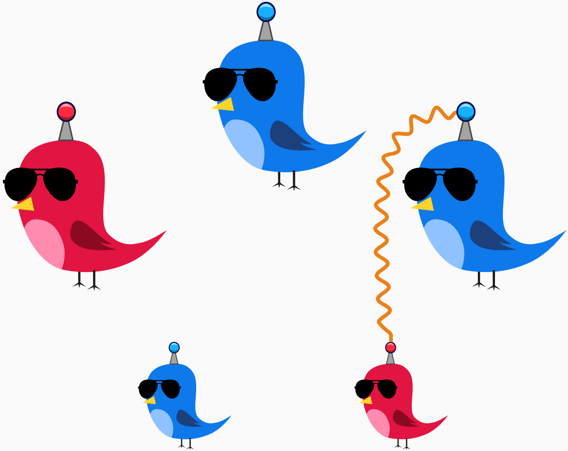
- Two large birds of different colors become small
- Large birds convert small birds to their color



More blue birds than red birds?

Protocol:

- Two large birds of different colors become small
- Large birds convert small birds to their color

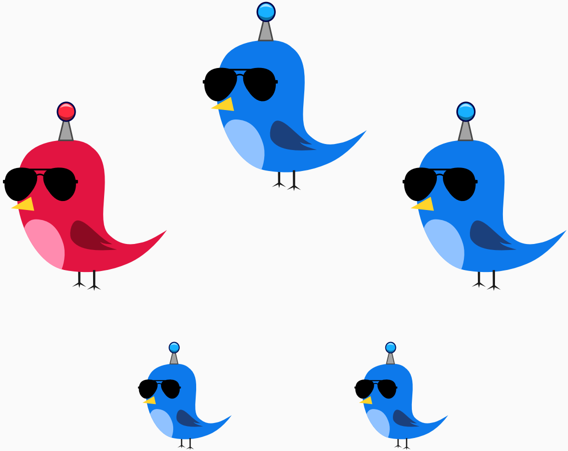


Example: majority protocol

More blue birds than red birds?

Protocol:

- Two large birds of different colors become small
- Large birds convert small birds to their color

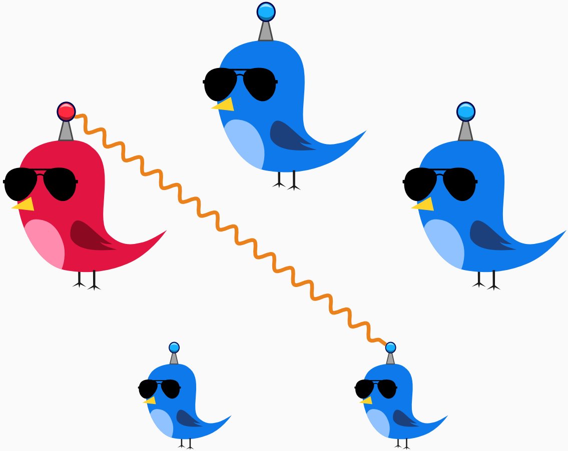


Example: majority protocol

More blue birds than red birds?

Protocol:

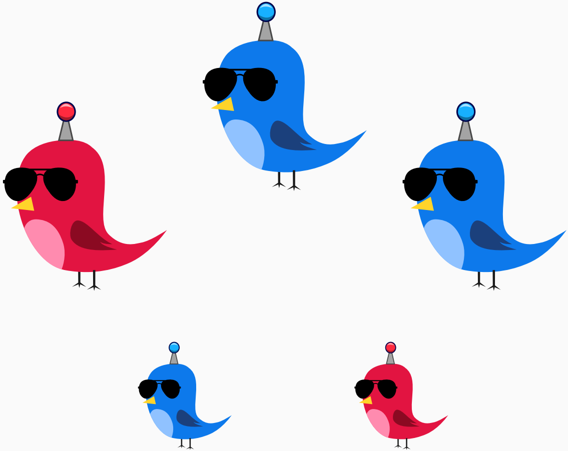
- Two large birds of different colors become small
- Large birds convert small birds to their color



More blue birds than red birds?

Protocol:

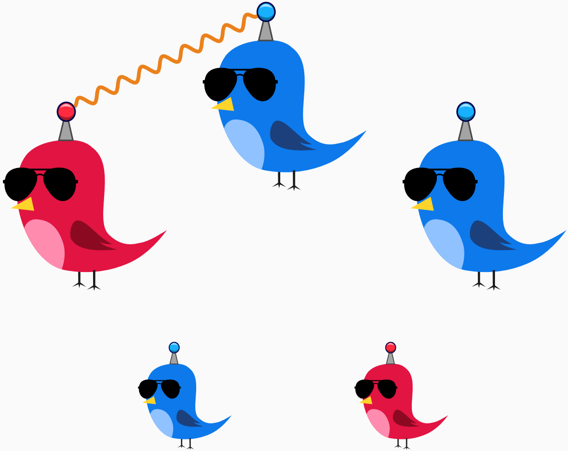
- Two large birds of different colors become small
- Large birds convert small birds to their color



More blue birds than red birds?

Protocol:

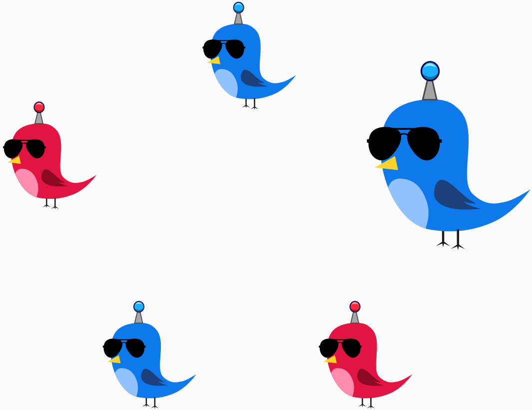
- Two large birds of different colors become small
- Large birds convert small birds to their color



More blue birds than red birds?

Protocol:

- Two large birds of different colors become small
- Large birds convert small birds to their color

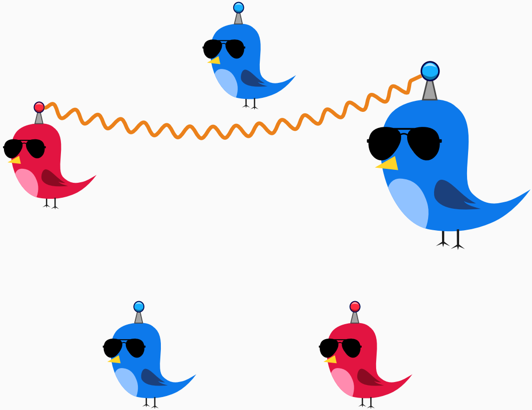


Example: majority protocol

More blue birds than red birds?

Protocol:

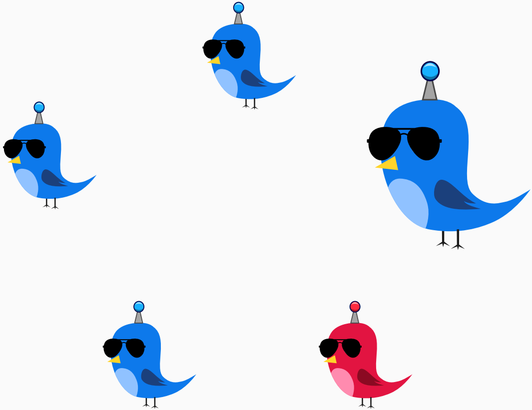
- Two large birds of different colors become small
- Large birds convert small birds to their color



More blue birds than red birds?

Protocol:

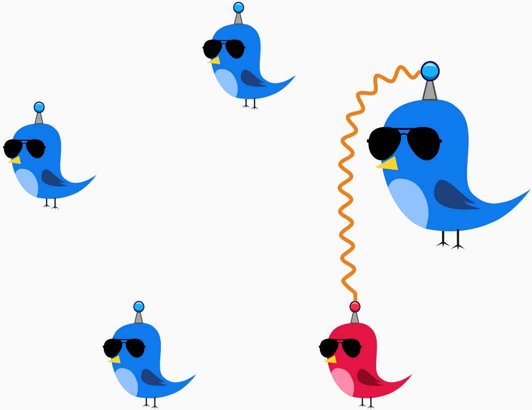
- Two large birds of different colors become small
- Large birds convert small birds to their color



More blue birds than red birds?

Protocol:

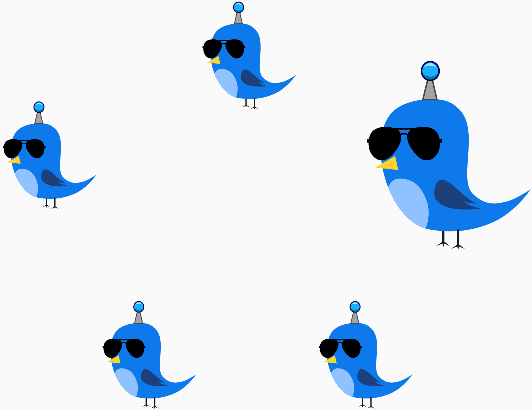
- Two large birds of different colors become small
- Large birds convert small birds to their color



More blue birds than red birds?

Protocol:

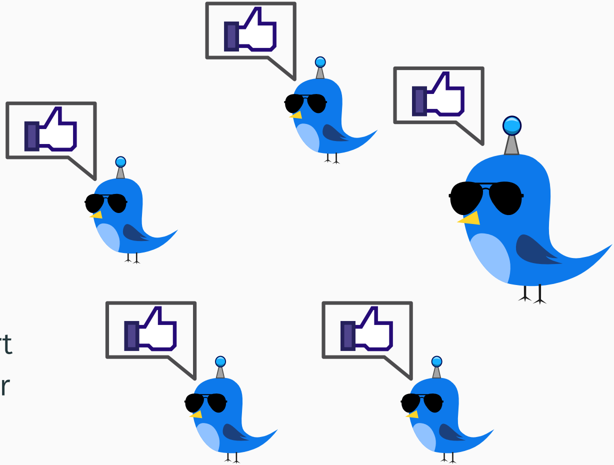
- Two large birds of different colors become small
- Large birds convert small birds to their color



More blue birds than red birds?

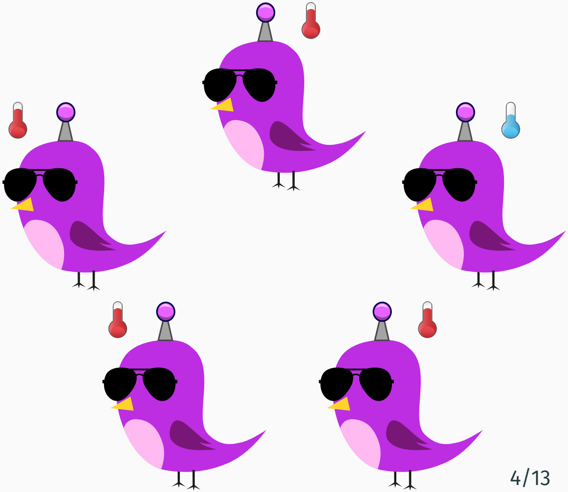
Protocol:

- Two large birds of different colors become small
- Large birds convert small birds to their color



Example: threshold protocol

Are there at least 4 sick birds?

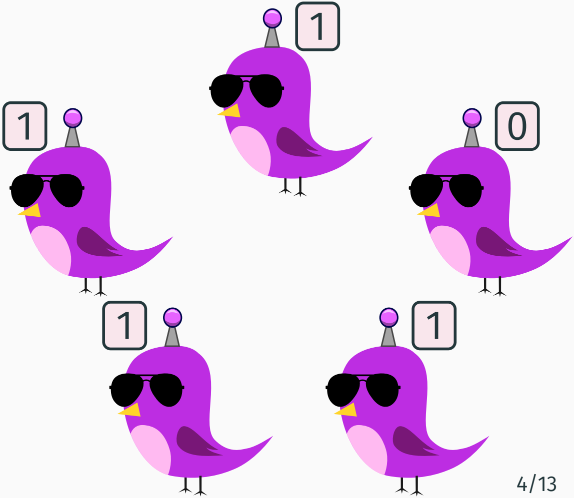


Example: threshold protocol

Are there at least 4 sick birds?

Protocol:

- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$

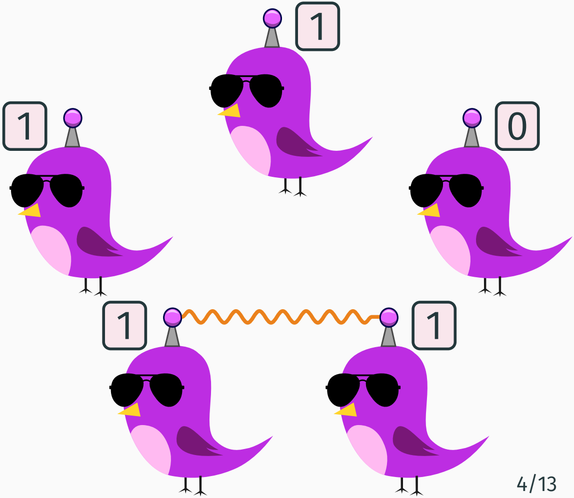


Example: threshold protocol

Are there at least 4 sick birds?

Protocol:

- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$

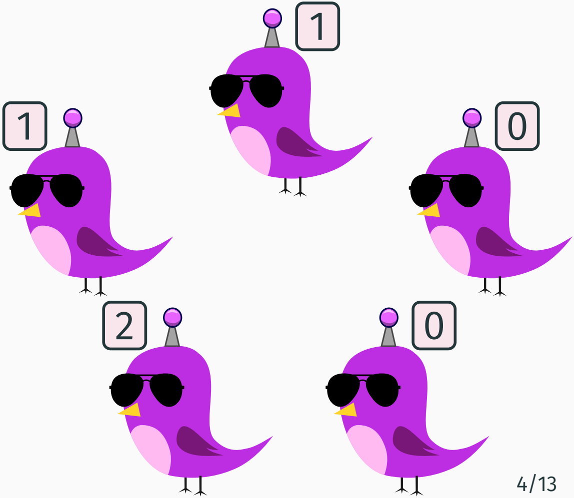


Example: threshold protocol

Are there at least 4 sick birds?

Protocol:

- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$

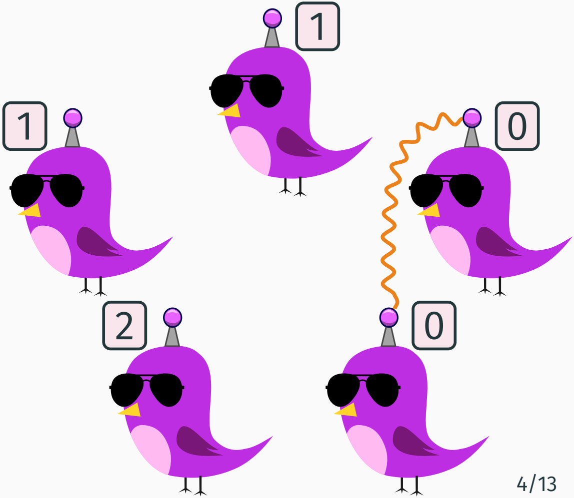


Example: threshold protocol

Are there at least 4 sick birds?

Protocol:

- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$

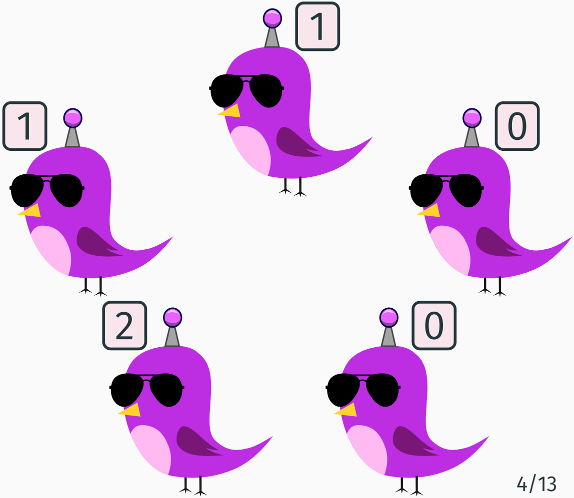


Example: threshold protocol

Are there at least 4 sick birds?

Protocol:

- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$

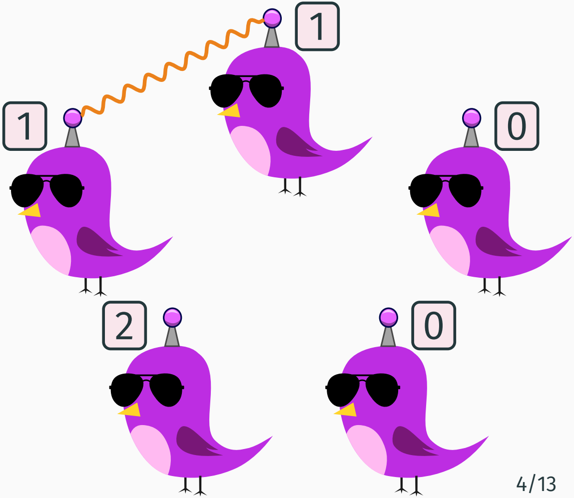


Example: threshold protocol

Are there at least 4 sick birds?

Protocol:

- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$

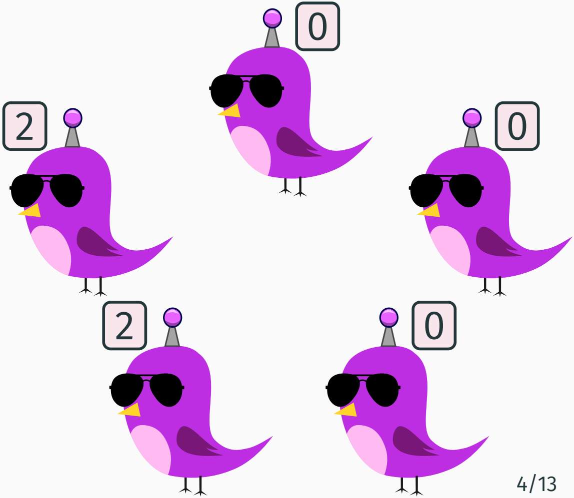


Example: threshold protocol

Are there at least 4 sick birds?

Protocol:

- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$

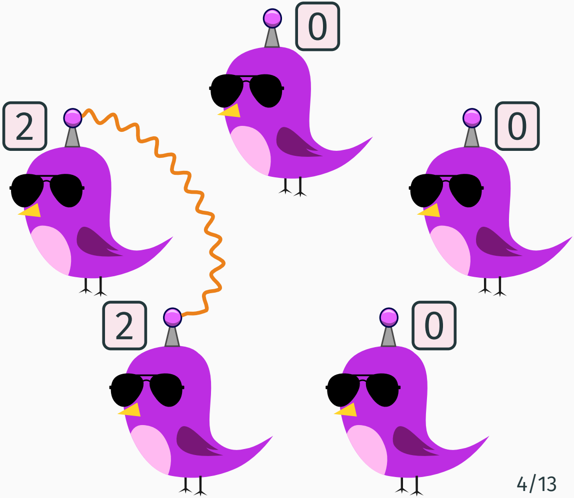


Example: threshold protocol

Are there at least 4 sick birds?

Protocol:

- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$

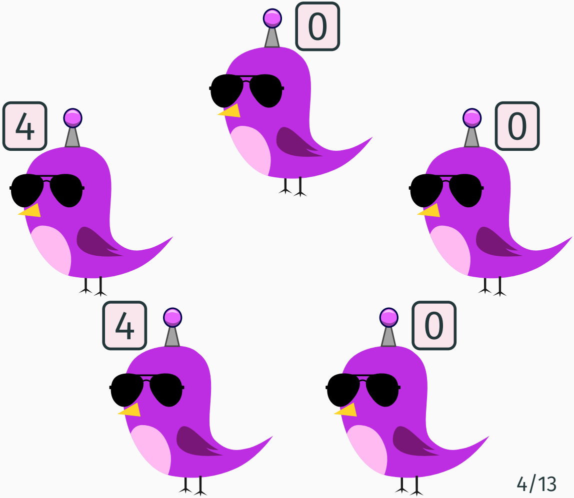


Example: threshold protocol

Are there at least 4 sick birds?

Protocol:

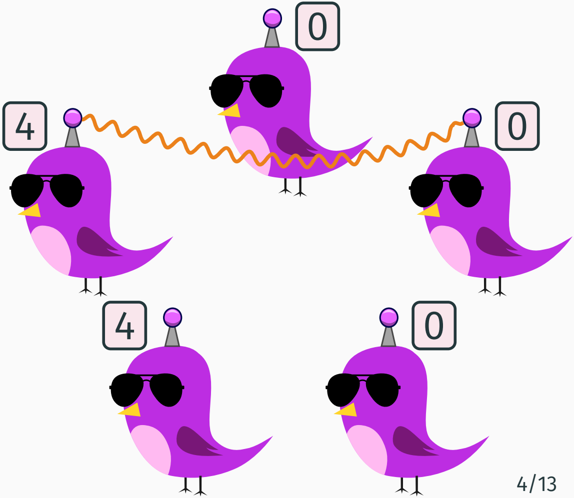
- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$



Are there at least 4 sick birds?

Protocol:

- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m + n, 0)$ if $m + n < 4$
- $(m, n) \mapsto (4, 4)$ if $m + n \geq 4$

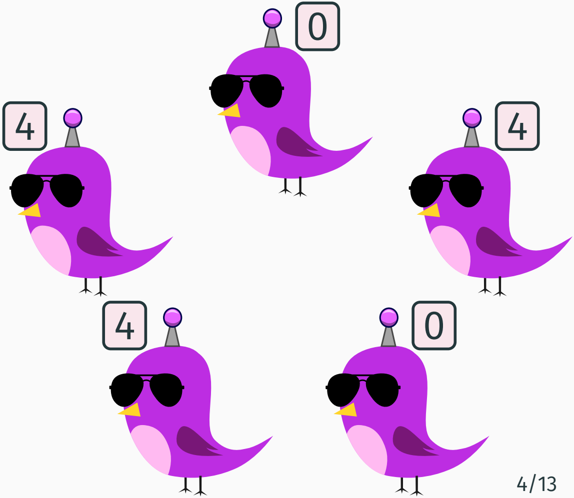


Example: threshold protocol

Are there at least 4 sick birds?

Protocol:

- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m + n, 0)$ if $m + n < 4$
- $(m, n) \mapsto (4, 4)$ if $m + n \geq 4$

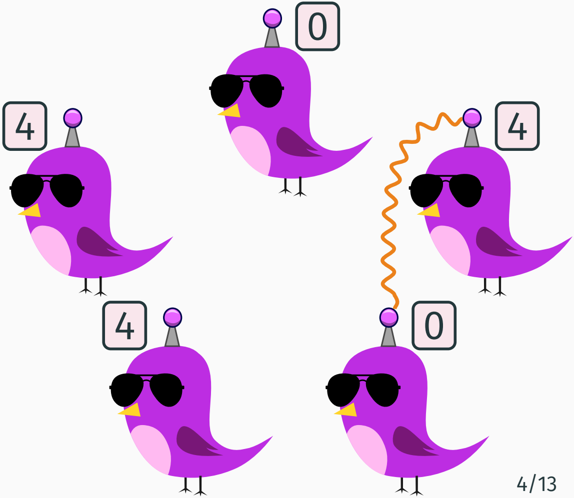


Example: threshold protocol

Are there at least 4 sick birds?

Protocol:

- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m + n, 0)$ if $m + n < 4$
- $(m, n) \mapsto (4, 4)$ if $m + n \geq 4$

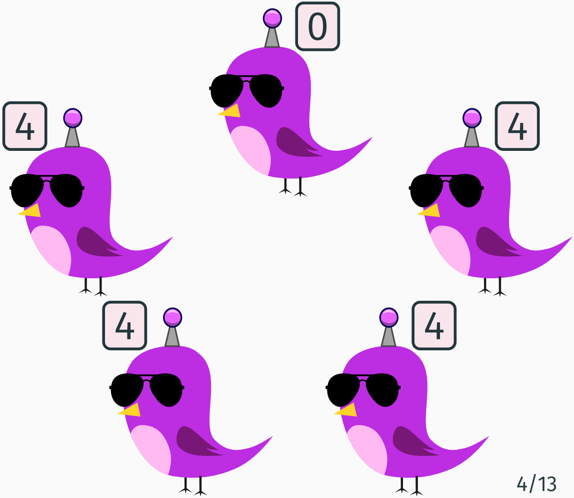


Example: threshold protocol

Are there at least 4 sick birds?

Protocol:

- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m+n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$

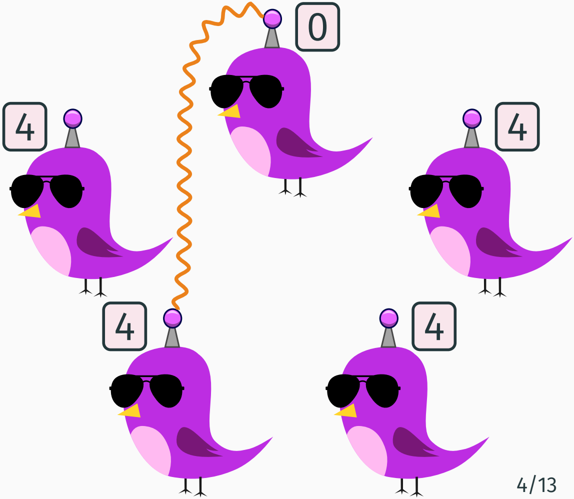


Example: threshold protocol

Are there at least 4 sick birds?

Protocol:

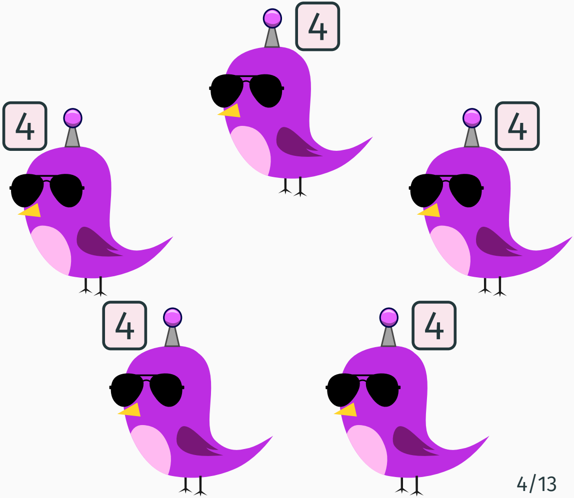
- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$



Are there at least 4 sick birds?

Protocol:

- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$

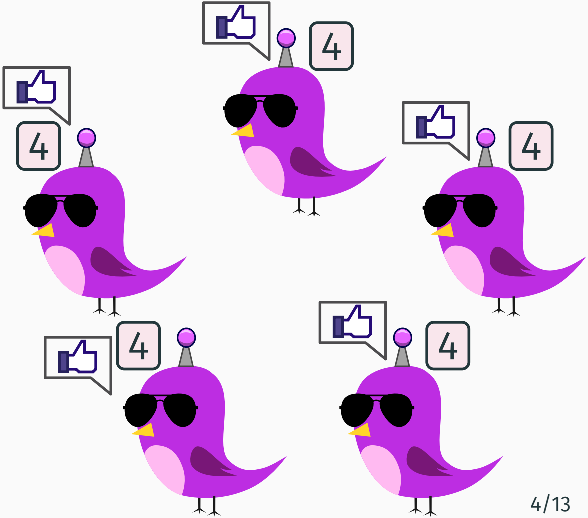


Example: threshold protocol

Are there at least 4 sick birds?

Protocol:

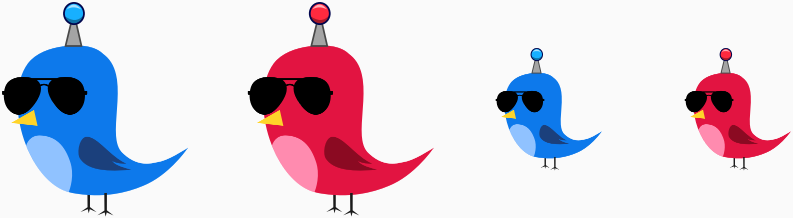
- Each bird is in a state of $\{0, 1, 2, 3, 4\}$
- Sick birds initially in state 1 and healthy birds in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$



Demonstration

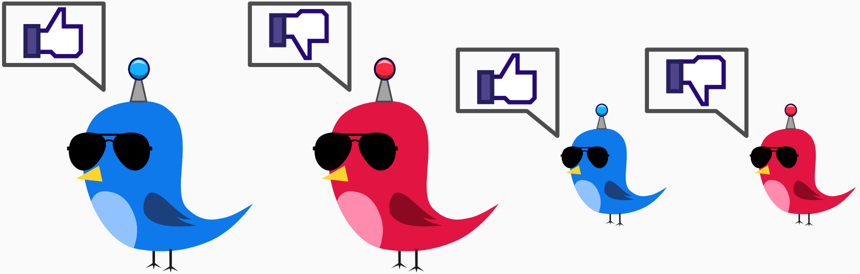
Population protocols: formal model

- *States*: finite set Q
- *Opinions*: $O : Q \to \{0, 1\}$
- *Initial states*: $I \subseteq Q$
- *Transitions*: $T \subseteq Q^2 \times Q^2$



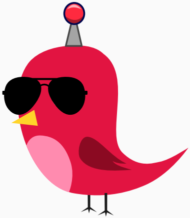
Population protocols: formal model

- *States*: finite set Q
- *Opinions*: $O : Q \to \{0, 1\}$
- *Initial states*: $I \subseteq Q$
- *Transitions*: $T \subseteq Q^2 \times Q^2$



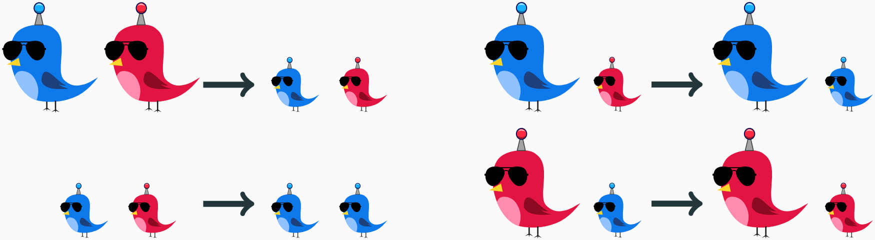
Population protocols: formal model

- *States*: finite set Q
- *Opinions*: $O : Q \to \{0, 1\}$
- *Initial states*: $I \subseteq Q$
- *Transitions*: $T \subseteq Q^2 \times Q^2$

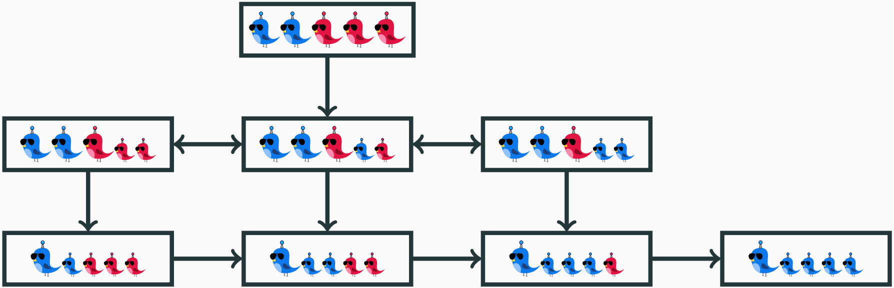



Population protocols: formal model

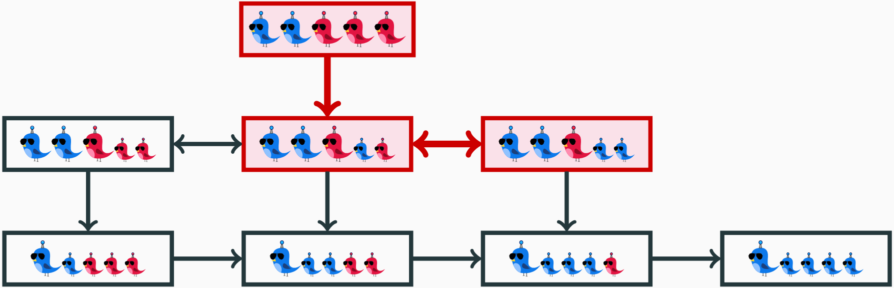
- *States*: finite set Q
- *Opinions*: $O : Q \to \{0, 1\}$
- *Initial states*: $I \subseteq Q$
- *Transitions*: $T \subseteq Q^2 \times Q^2$



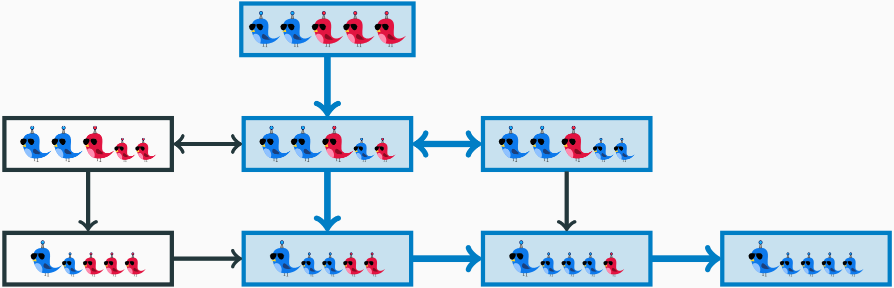
Reachability graph:



Executions must be fair:

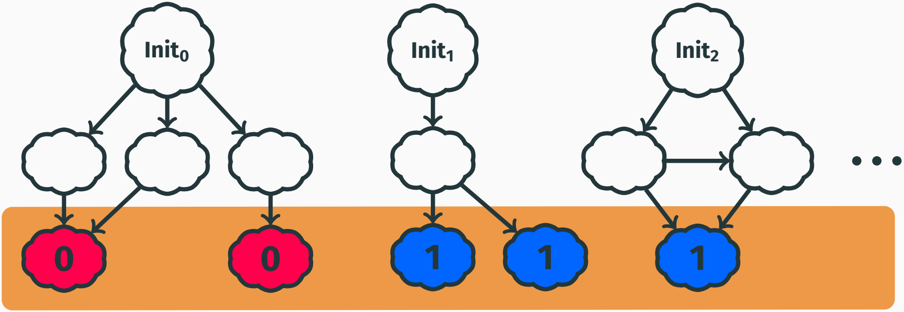


Executions must be fair:



A protocol computes a predicate $f\colon \mathbb{N}^I \to \{0,1\}$

if fair executions reach common **consensus**



Population protocols: formal model

A protocol computes a predicate $f\colon \mathbb{N}^I \to \{0,1\}$

if fair executions reach common **consensus**

Expressive power **Angluin, Aspnes, Eisenstat PODC'06**

Population protocols compute precisely predicates definable in Presburger arithmetic, *i.e.* FO($\mathbb{N}$, $+$, $<$)

Protocols can become complex, even for B ≥ R:

Fast and Exact Majority in Population Protocols

Dan Alistarh
Microsoft Research

Rati Gelashvili*
MIT

Milan Vojnović
Microsoft Research

$weight(x) = \begin{cases} |x| & \text{if } x \in StrongStates \text{ or } x \in WeakStates; \\ 1 & \text{if } x \in IntermediateStates. \end{cases}$

$sgn(x) = \begin{cases} 1 & \text{if } x \in \{+0, 1_d, \ldots, 1_1, 3, 5, \ldots, m\}; \\ -1 & \text{otherwise.} \end{cases}$

$value(x) = sgn(x) \cdot weight(x)$

```
/* Functions for rounding state interactions */
```

$\phi(x) = -1_1$ if $x = -1; 1_1$ if $x = 1; x,$ otherwise

$R_\downarrow(k) = \phi(k$ if k odd integer, $k-1$ if k even$)$

$R_\uparrow(k) = \phi(k$ if k odd integer, $k+1$ if k even$)$

$Shift\text{-}to\text{-}Zero(x) = \begin{cases} -1_{j+1} & \text{if } x = -1_j \text{ for some index } j < d \\ 1_{j+1} & \text{if } x = 1_j \text{ for some index } j < d \\ x & \text{otherwise.} \end{cases}$

$Sign\text{-}to\text{-}Zero(x) = \begin{cases} +0 & \text{if } sgn(x) > 0 \\ -0 & \text{oherwise.} \end{cases}$

procedure update$\langle x, y \rangle$
  if $(weight(x) > 0$ *and* $weight(y) > 1)$ ***or*** $(weight(y) > 0$ *and* $weight(x) > 1)$ **then**
    $x' \leftarrow R_\downarrow\left(\frac{value(x)+value(y)}{2}\right)$ and $y' \leftarrow R_\uparrow\left(\frac{value(x)+value(y)}{2}\right)$
  else if $weight(x) \cdot weight(y) = 0$ *and* $value(x) + value(y) > 0$ **then**
    if $weight(x) \neq 0$ **then** $x' \leftarrow Shift\text{-}to\text{-}Zero(x)$ and $y' \leftarrow Sign\text{-}to\text{-}Zero(x)$
    else $y' \leftarrow Shift\text{-}to\text{-}Zero(y)$ and $x' \leftarrow Sign\text{-}to\text{-}Zero(y)$
  else if $(x \in \{-1_d, +1_d\}$ *and* $weight(y) = 1$ *and* $sgn(x) \neq sgn(y))$ ***or***
      $(y \in \{-1_d, +1_d\}$ *and* $weight(x) = 1$ *and* $sgn(y) \neq sgn(x))$ **then**
    $x' \leftarrow -0$ and $y' \leftarrow +0$
  else
    $x' \leftarrow Shift\text{-}to\text{-}Zero(x)$ and $y' \leftarrow Shift\text{-}to\text{-}Zero(y)$

Protocols can become complex, even for B ≥ R:

Fast and Exact Majority in Population Protocols

Dan Alistarh
Microsoft Research

Rati Gelashvili*
MIT

Milan Vojnović
Microsoft Research

$weight(x) = \begin{cases} |x| & \text{if } x \in StrongStates \text{ or } x \in WeakStates; \\ 1 & \text{if } x \in IntermediateStates. \end{cases}$

$sgn(x) = \begin{cases} 1 & \text{if } x \in \{+0, 1_d, \ldots, 1_1, 3, 5, \ldots, m\}; \\ -1 & \text{otherwise.} \end{cases}$

$value(x) = sgn(x) \cdot weight(x)$

/* Functions for rounding state interactions */

$\phi(x) = -1_1$ if $x = -1; 1_1$ if $x = 1; x$, otherwise

$R_\downarrow(k) = \phi(k$ if k odd integer, $k-1$ if k even)

$R_\uparrow(k) = \phi(k$ if k odd integer, $k+1$ if k even)

$Shift\text{-}to\text{-}Zero(x) = \begin{cases} -1_{j+1} & \text{if } x = -1_j \text{ for some index } j < d \\ 1_{j+1} & \text{if } x = 1_j \text{ for some index } j < d \\ x & \text{otherwise.} \end{cases}$

$Sign\text{-}to\text{-}Zero(x) = \begin{cases} +0 & \text{if } sgn(x) > 0 \\ -0 & \text{oherwise.} \end{cases}$

procedure update$\langle x, y \rangle$
    if $(weight(x) > 0$ *and* $weight(y) > 1)$ ***or*** $(weight(y) > 0$ *and* $weight(x) > 1)$ **then**
        $x' \leftarrow R_\downarrow\left(\frac{value(x)+value(y)}{2}\right)$ and $y' \leftarrow R_\uparrow\left(\frac{value(x)+value(y)}{2}\right)$
    else if $weight(x) \cdot weight(y) = 0$ *and* $value(x) + value(y) > 0$ **then**
        if $weight(x) \neq 0$ **then** $x' \leftarrow Shift\text{-}to\text{-}Zero(x)$ and $y' \leftarrow Sign\text{-}to\text{-}Zero(x)$
        else $y' \leftarrow Shift\text{-}to\text{-}Zero(y)$ and $x' \leftarrow Sign\text{-}to\text{-}Zero(y)$
    else if $(x \in \{-1_d, +1_d\}$ *and* $weight(y) = 1$ *and* $sgn(x) \neq sgn(y))$ ***or***
        $(y \in \{-1_d, +1_d\}$ *and* $weight(x) = 1$ *and* $sgn(y) \neq sgn(x))$ **then**
        $x' \leftarrow -0$ and $y' \leftarrow +0$
    else
        $x' \leftarrow Shift\text{-}to\text{-}Zero(x)$ and $y' \leftarrow Shift\text{-}to\text{-}Zero(y)$

How to verify correctness automatically?

Analysis of protocols

Number of states corresponds to amount of memory, relevant to keep it minimal for embedded systems

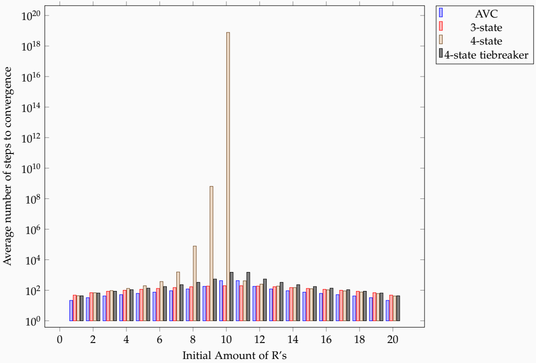
- **B ≥ R** requires at least 4 states (Mertzios *et al.* ICALP'14)
- **X ≥ c** requires at most c + 1 states

Number of states corresponds to amount of memory, relevant to keep it minimal for embedded systems

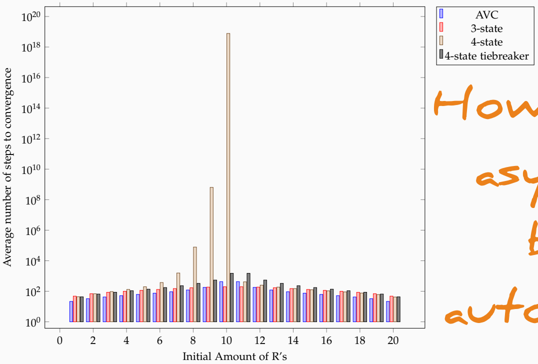
- **B ≥ R** requires at least 4 states (Mertzios *et al.* ICALP'14)
- **X ≥ c** requires at most c + 1 states

What is the state complexity of common predicates?

Convergence speed may vary wildly, challenging to establish bounds



Convergence speed may vary wildly, challenging to establish bounds



How to derive asymptotic bounds automatically?

1. **Automatic verification of correctness**
   - PODC'17 with Javier, Stefan and Philipp
   - Submission to CAV'18 with Javier and Stefan
   - Interns: Philip Offtermatt and Amrita Suresh
2. **State complexity of common predicates**
   - STACS'18 with Javier and Stefan
3. **Automatic analysis of convergence speed**
   - Ongoing work with Javier and Antonín Kučera

Analysis of protocols

1. **Automatic verification of correctness**
   - PODC'17 with Javier, Stefan and Philipp
   - Submission to CAV'18 with Javier and Stefan

2. **State complexity of common predicates**
   - STACS'18 with Javier and Stefan

This talk

Existing verification tools:

- PAT: model checker with global fairness

  (Sun, Liu, Song Dong and Pang CAV'09)

- `bp-ver`: graph exploration

  (Chatzigiannakis, Michail and Spirakis SSS'10)

- Conversion to counter machines + `PRISM`/`Spin`

  (Clément, Delporte-Gallet, Fauconnier and Sighireanu ICDCS'11)

Verification: state of the art

Existing verification tools:

- PAT: model checker with global fairness

  (Sun, Liu, Song Dong and Pang CAV'09)

- `bp-ver`: graph exploration

  (Chatzigiannakis, Michail and Spirakis SSS'10)

- Conversion to counter machines + `PRISM`/`Spin`

  (Clément, Delporte-Gallet, Fauconnier and Sighireanu ICDCS'11)

Only for populations of fixed size!

Verification: state of the art

Sometimes possible to verify all sizes:

- Verification with the interactive theorem prover Coq

(Deng and Monin TASE'09)

Sometimes possible to verify all sizes:

- Verification with the interactive theorem prover Coq

(Deng and Monin TASE'09)

Not automatic!

Sometimes possible to verify all sizes:

- Verification with the interactive theorem prover Coq

(Deng and Monin TASE'09)

Challenge: verifying automatically all sizes

Testing whether a protocol computes φ amounts to testing:

$$\neg\exists C, D: \quad C \xrightarrow{*} D \wedge$$
$$C \text{ is initial } \wedge$$
$$D \text{ is in a BSCC} \wedge$$
$$\text{opinion}(D) \neq \varphi(C)$$

Testing whether a protocol computes φ amounts to testing:

$$\neg\exists C, D: \quad \begin{array}{l} C \xrightarrow{*} D \wedge \\ C \text{ is initial } \wedge \\ D \text{ is in a BSCC} \wedge \\ \text{opinion}(D) \neq \varphi(C) \end{array}$$

As difficult as verification

Testing whether a protocol computes φ amounts to testing:

$$\neg\exists C, D: \quad C \overset{*}{\dashrightarrow} D \wedge$$
$$C \text{ is initial} \wedge$$
$$D \text{ is in a BSCC} \wedge$$
$$\text{opinion}(D) \neq \varphi(C)$$

Relaxed with Presburger-definable overapproximation!

Testing whether a protocol computes φ amounts to testing:

$$\neg\exists C, D: \quad C \overset{*}{\dashrightarrow} D \;\wedge$$
$$C \text{ is initial } \wedge$$
$$D \text{ is in a BSCC} \wedge$$
$$\text{opinion}(D) \neq \varphi(C)$$

Difficult to express

Testing whether a protocol computes φ amounts to testing:

$$\neg\exists C, D: \quad C \overset{*}{\dashrightarrow} D \wedge$$
$$C \text{ is initial} \wedge$$
$$D \text{ is terminal} \wedge$$
$$\text{opinion}(D) \neq \varphi(C)$$

BSCCs are of size 1 for most protocols!

Testing whether a protocol computes φ amounts to testing:

$$\neg\exists C, D: \quad \begin{aligned} & C \overset{*}{\dashrightarrow} D \wedge \\ & C \text{ is initial} \wedge \\ & D \text{ is terminal} \wedge \\ & \text{opinion}(D) \neq \varphi(C) \end{aligned}$$

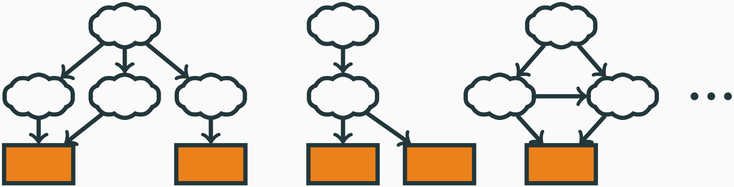
Testable with an SMT solver

Testing whether a protocol computes φ amounts to testing:

$$\begin{aligned}\neg\exists C, D: \quad & C \overset{*}{\dashrightarrow} D \wedge \\ & C \text{ is initial} \wedge \\ & D \text{ is terminal } \wedge \\ & \text{opinion}(D) \neq \varphi(C)\end{aligned}$$

But how to know whether all BSCCs are of size 1?

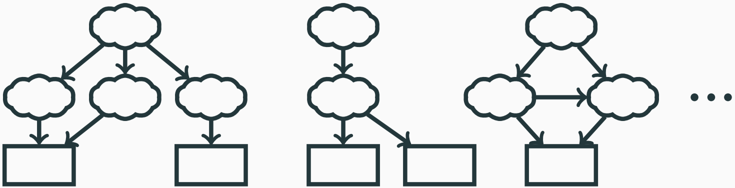
Protocol is *silent* if fair executions reach terminal configurations



BSCCs of size 1

Protocol is *silent* if fair executions reach terminal configurations

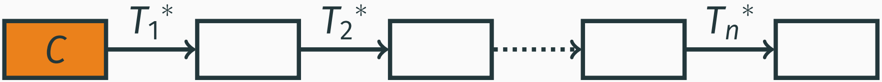
- Testing silentness is as hard as verification of correctness
- But most protocols satisfy a common design



BSCCs of size 1

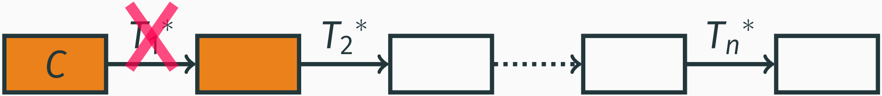
Partition $T = T_1 \cup T_2 \cup \cdots \cup T_n$ s.t. for every i

- all executions restricted to T_i terminate
- if $T_1 \cup \cdots \cup T_{i-1}$ disabled in C and $C \xrightarrow{T_i{}^*} D$, then $T_1 \cup \cdots \cup T_{i-1}$ also disabled in D



Partition $T = T_1 \cup T_2 \cup \cdots \cup T_n$ s.t. for every i

- all executions restricted to T_i terminate
- if $T_1 \cup \cdots \cup T_{i-1}$ disabled in C and $C \xrightarrow{T_i^*} D$, then $T_1 \cup \cdots \cup T_{i-1}$ also disabled in D



Partition $T = T_1 \cup T_2 \cup \cdots \cup T_n$ s.t. for every i

- all executions restricted to T_i terminate
- if $T_1 \cup \cdots \cup T_{i-1}$ disabled in C and $C \xrightarrow{T_i{}^*} D$, then $T_1 \cup \cdots \cup T_{i-1}$ also disabled in D

Partition $T = T_1 \cup T_2 \cup \cdots \cup T_n$ s.t. for every i

- all executions restricted to T_i terminate
- if $T_1 \cup \cdots \cup T_{i-1}$ disabled in C and $C \xrightarrow{T_i{}^*} D$, then $T_1 \cup \cdots \cup T_{i-1}$ also disabled in D

Common design: layered termination

Partition $T = T_1 \cup T_2 \cup \cdots \cup T_n$ s.t. for every i

- all executions restricted to T_i terminate
- if $T_1 \cup \cdots \cup T_{i-1}$ disabled in C and $C \xrightarrow{T_i{}^*} D$, then $T_1 \cup \cdots \cup T_{i-1}$ also disabled in D

T_1

$$\begin{aligned} B\ R &\rightarrow b\ r \\ R\ b &\rightarrow R\ r \\ B\ r &\rightarrow B\ b \\ b\ r &\rightarrow b\ b \end{aligned}$$

T_1

$$\mathbf{B}\ \mathbf{R} \to \mathbf{b}\ \mathbf{r}$$
$$\mathbf{R}\ \mathbf{b} \to \mathbf{R}\ \mathbf{r}$$
$$\mathbf{B}\ \mathbf{r} \to \mathbf{B}\ \mathbf{b}$$
$$\mathbf{b}\ \mathbf{r} \to \mathbf{b}\ \mathbf{b}$$

Bad partition: not all executions over T_1 terminate

T_1

$$\boldsymbol{B}\ \boldsymbol{R} \to \boldsymbol{b}\ \boldsymbol{r}$$
$$\boxed{\begin{array}{l}\boldsymbol{R}\ \boldsymbol{b} \to \boldsymbol{R}\ \boldsymbol{r} \\ \boldsymbol{B}\ \boldsymbol{r} \to \boldsymbol{B}\ \boldsymbol{b}\end{array}}$$
$$\boldsymbol{b}\ \boldsymbol{r} \to \boldsymbol{b}\ \boldsymbol{b}$$

Bad partition: not all executions over T_1 terminate

$$\{\boldsymbol{B}, \boldsymbol{B}, \boldsymbol{R}, \boldsymbol{R}\} \to \{\boldsymbol{B}, \boldsymbol{b}, \boldsymbol{r}, \boldsymbol{R}\} \to \{\boldsymbol{B}, \boldsymbol{b}, \boldsymbol{b}, \boldsymbol{R}\} \to$$
$$\{\boldsymbol{B}, \boldsymbol{b}, \boldsymbol{r}, \boldsymbol{R}\} \to \{\boldsymbol{B}, \boldsymbol{b}, \boldsymbol{b}, \boldsymbol{R}\} \to \cdots$$

T_1

$$\mathbf{B}\ \mathbf{R} \to \mathbf{b}\ \mathbf{r}$$

T_2

$$\mathbf{R}\ \mathbf{b} \to \mathbf{R}\ \mathbf{r}$$

T_3

$$\mathbf{B}\ \mathbf{r} \to \mathbf{B}\ \mathbf{b}$$

$$\mathbf{b}\ \mathbf{r} \to \mathbf{b}\ \mathbf{b}$$

T_1	T_2	T_3
$B\ R \to b\ r$	$R\ b \to R\ r$	$B\ r \to B\ b$ $b\ r \to b\ b$

$\#B \geq \#R$:

$$\{B^*,\ R^*\}$$

T_1 ✗

$$B\,R \to b\,r$$

T_2

$$R\,b \to R\,r$$

T_3

$$B\,r \to B\,b$$
$$b\,r \to b\,b$$

$\#B \geq \#R$:

$$\{B^*,\ R^*\} \xrightarrow{*} \{B^*,\ b^*,\ r^*\}$$

T_1 ✗

$B\,R \to b\,r$

T_2 ✗

$R\,b \to R\,r$

T_3

$B\,r \to B\,b$

$b\,r \to b\,b$

$\#B \geq \#R$:

$$\{B^*,\ R^*\} \xrightarrow{*} \{B^*,\ b^*,\ r^*\}$$

T_1 ✗

$\mathbf{B}\,\mathbf{R} \rightarrow \mathbf{b}\,\mathbf{r}$

T_2 ✗

$\mathbf{R}\,\mathbf{b} \rightarrow \mathbf{R}\,\mathbf{r}$

T_3 ✗

$\mathbf{B}\,\mathbf{r} \rightarrow \mathbf{B}\,\mathbf{b}$

$\mathbf{b}\,\mathbf{r} \rightarrow \mathbf{b}\,\mathbf{b}$

$\#\mathbf{B} \geq \#\mathbf{R}$:

$$\{\mathbf{B}^*,\ \mathbf{R}^*\} \xrightarrow{*} \{\mathbf{B}^*,\ \mathbf{b}^*,\ \mathbf{r}^*\} \xrightarrow{*} \{\mathbf{B}^*,\ \mathbf{b}^*\}$$

T_1

$$B\,R \rightarrow b\,r$$

T_2

$$R\,b \rightarrow R\,r$$

T_3

$$B\,r \rightarrow B\,b$$
$$b\,r \rightarrow b\,b$$

$\#B \geq \#R$:

$$\{B^*,\ R^*\} \xrightarrow{*} \{B^*,\ b^*,\ r^*\} \xrightarrow{*} \{B^*,\ b^*\}$$

$\#R > \#B$:

$$\{R^+,\ B^*\}$$

T_1 ✗

$$B\,R \to b\,r$$

T_2

$$R\,b \to R\,r$$

T_3

$$B\,r \to B\,b$$
$$b\,r \to b\,b$$

$\#B \geq \#R$:

$$\{B^*,\ R^*\} \xrightarrow{*} \{B^*,\ b^*,\ r^*\} \xrightarrow{*} \{B^*,\ b^*\}$$

$\#R > \#B$:

$$\{R^+,\ B^*\} \xrightarrow{*} \{R^+,\ r^*,\ b^*\}$$

T_1 ✗

$B\ R \to b\ r$

T_2 ✗

$R\ b \to R\ r$

T_3

$B\ r \to B\ b$

$b\ r \to b\ b$

$\#B \geq \#R$:

$$\{B^*,\ R^*\} \xrightarrow{*} \{B^*,\ b^*,\ r^*\} \xrightarrow{*} \{B^*,\ b^*\}$$

$\#R > \#B$:

$$\{R^+,\ B^*\} \xrightarrow{*} \{R^+,\ r^*,\ b^*\} \xrightarrow{*} \{R^+,\ r^*\}$$

T_1 ✗

$B\ R \to b\ r$

T_2 ✗

$R\ b \to R\ r$

T_3 ✗

$B\ r \to B\ b$

$b\ r \to b\ b$

$\#B \geq \#R$:

$$\{B^*,\ R^*\} \xrightarrow{*} \{B^*,\ b^*,\ r^*\} \xrightarrow{*} \{B^*,\ b^*\}$$

$\#R > \#B$:

$$\{R^+,\ B^*\} \xrightarrow{*} \{R^+,\ r^*,\ b^*\} \xrightarrow{*} \{R^+,\ r^*\}$$

Theorem PODC'17

Deciding whether a protocol is strongly silent $\in$ NP

Proof sketch

Guess partition $T = T_1 \cup T_2 \cup \cdots \cup T_n$ and test whether it is correct by verifying

- Petri net structural termination
- Additional simple structural properties

Theorem PODC'17

Strongly silent protocols as expressive as general protocols

Proof sketch

- Protocols for

$$a_1x_1 + \ldots + a_nx_n \geq b$$

$$a_1x_1 + \ldots + a_nx_n \equiv b \pmod{m}$$

  have layered termination partitions

- Conjunction and negation preserve layered termination

Peregrine: Haskell + Z3 + JavaScript (front end)

gitlab.lrz.de/i7/peregrine

Protocol	Predicate	# states	# trans.	Time (secs.)
Majority [a]	$x \geq y$	4	4	0.1
Broadcast [b]	$x_1 \vee \cdots \vee x_n$	2	1	0.1
Linear ineq. [c]	$\sum a_i x_i \geq 9$	75	2148	2376
Modulo [c]	$\sum a_i x_i = 0 \bmod 70$	72	2555	3177
Threshold [d]	$x \geq 50$	51	1275	182
Threshold [b]	$x \geq 325$	326	649	3471
Threshold [e]	$x \geq 10^7$	37	155	19

[a] Draief *et al.* 2012
[b] Clément *et al.* 2011
[c] Angluin *et al.* 2006
[d] Chatzigiannakis *et al.* 2010
[e] Offtermatt 2017 (bachelor thesis)

Demonstration

Threshold state complexity: logarithmic bounds

Given: Presburger-definable predicate φ

Question: Smallest number of states necessary to compute φ?

Threshold state complexity: logarithmic bounds

Given: Presburger-definable predicate φ

Question: Smallest number of states necessary to compute φ?

Difficult problem...
What about basic predicates?

Threshold state complexity: logarithmic bounds

Given: $c \in \mathbb{N}$

Question: Smallest number of states necessary to compute $x \geq c$?

Threshold state complexity: logarithmic bounds

Given: $c \in \mathbb{N}$

Question: Smallest number of states necessary to compute $x \geq c$?

Upper bound: $c + 1$

Lower bound: 2

Threshold state complexity: logarithmic bounds

Given: $c \in \mathbb{N}$

Question: Smallest number of states necessary to compute $x \geq c$?

Upper bound: $c + 1$

Lower bound: 2

Theorem STACS'18

Computable with $O(\log c)$ states, if $c = 2^n$.

Proof sketch

$$
\begin{array}{ccc}
(1,1) & \mapsto & (2,0) \\
(2,2) & \mapsto & (4,0) \\
\vdots & & \vdots \\
(2^{n-1}, 2^{n-1}) & \mapsto & (2^n, 0) \\
(2^n, m) & \mapsto & (2^n, 2^n)
\end{array}
$$

Threshold state complexity: logarithmic bounds

Given: $c \in \mathbb{N}$

Question: Smallest number of states necessary to compute $x \geq c$?

Upper bound: $c + 1$

Lower bound: 2

Theorem STACS'18

Computable with $O(\log c)$ states, ~~if $c = 2^n$~~.

Proof sketch

$$
\begin{array}{ccc}
(1, 1) & \mapsto & (2, 0) \\
(2, 2) & \mapsto & (4, 0) \\
\vdots & & \vdots \\
(2^{n-1}, 2^{n-1}) & \mapsto & (2^n, 0) \\
(2^n, m) & \mapsto & (2^n, 2^n)
\end{array}
$$

+ extra states and transitions

Threshold state complexity: logarithmic bounds

Given: $c \in \mathbb{N}$

Question: Smallest number of states necessary to compute $x \geq c$?

Upper bound: $O(\log c)$

Lower bound: 2

Threshold state complexity: logarithmic bounds

Given: $c \in \mathbb{N}$

Question: Smallest number of states necessary to compute $x \geq c$?

Upper bound: $O(\log c)$

Lower bound: 2

Theorem **STACS'18**

Let $P_0, P_1, \ldots$ be protocols such that P_c computes $x \geq c$.
There are infinitely many c s.t. P_c has $\geq (\log c)^{1/4}$ states.

Proof sketch

Counting argument on # unary predicates vs. # protocols.

Threshold state complexity: logarithmic bounds

Given: $c \in \mathbb{N}$

Question: Smallest number of states necessary to compute $x \geq c$?

Upper bound: $O(\log c)$

Lower bound: $\underbrace{O(\log^{1/4} c)}_{\text{for inf. many c}}$

Threshold state complexity: logarithmic bounds

Given: $c \in \mathbb{N}$

Question: Smallest number of states necessary to compute $x \geq c$?

Upper bound: $O(\log c)$

Lower bound: $\underbrace{O(\log^{1/4} c)}_{\text{for inf. many c}}$

Possible to go below log c for some c?

Threshold state complexity: logarithmic bounds

Given: $c \in \mathbb{N}$

Question: Smallest number of states necessary to compute $x \geq c$?

Upper bound: $O(\log c)$

Lower bound: $\underbrace{O(\log^{1/4} c)}_{\text{for inf. many c}}$

Possible to go below log c for some c?

Yes!

Threshold state complexity: sublogarithmic bounds

Theorem — **STACS'18**

There exist protocols $P_0, P_1, \ldots$ and numbers $c_0 < c_1 < \cdots$ such that P_i computes $x \geq c_i$ and has $O(\log \log c_i)$ states.

Threshold state complexity: sublogarithmic bounds

Theorem **STACS'18**

There exist protocols $P_0, P_1, \ldots$ and numbers $c_0 < c_1 < \cdots$ such that P_i computes $x \geq c_i$ and has $O(\log \log c_i)$ states.

Lemma **Mayr and Meyer '82**

For every $c \in \mathbb{N}$, there exists a reversible multiset rewriting system $\mathcal{R}_c$ over alphabet $\Sigma \supseteq \{x, y, z, w\}$ of size $O(c)$ with rewriting rules $T \subseteq \Sigma^{\leq 5} \times \Sigma^{\leq 5}$ such that

$$\{x, y\} \xrightarrow{*} M \text{ and } w \in M \iff M = \{y, z^{2^{2^c}}, w\}$$

Threshold state complexity: sublogarithmic bounds

Theorem **STACS'18**

There exist protocols $P_0, P_1, \ldots$ and numbers $c_0 < c_1 < \cdots$ such that P_i computes $x \geq c_i$ and has $O(\log \log c_i)$ states.

Proof sketch

- $\mathcal{R}_c$ can be simulated by adding a padding symbol $\perp$

Threshold state complexity: sublogarithmic bounds

Theorem **STACS'18**

There exist protocols $P_0, P_1, \ldots$ and numbers $c_0 < c_1 < \cdots$ such that P_i computes $x \geq c_i$ and has $O(\log \log c_i)$ states.

Proof sketch

- $\mathcal{R}_c$ can be simulated by adding a padding symbol $\perp$

Rewriting system $\mathcal{R}_c$	5-way population protocol
$(e, f, g) \mapsto (h, i)$	$(e, f, g, \perp, \perp) \mapsto (h, i, \perp, \perp, \perp)$
$(e, f) \mapsto (g, h, i)$	$(e, f, \perp, \perp, \perp) \mapsto (g, h, i, \perp, \perp)$

Threshold state complexity: sublogarithmic bounds

Theorem **STACS'18**

There exist protocols $P_0, P_1, \ldots$ and numbers $c_0 < c_1 < \cdots$ such that P_i computes $x \geq c_i$ and has $O(\log \log c_i)$ states.

Proof sketch

- $\mathcal{R}_c$ can be simulated by adding a padding symbol $\perp$

Each 5-way transition is converted to a "gadget" of 2-way transitions

Threshold state complexity: sublogarithmic bounds

Theorem **STACS'18**

There exist protocols $P_0, P_1, \ldots$ and numbers $c_0 < c_1 < \cdots$ such that P_i computes $x \geq c_i$ and has $O(\log \log c_i)$ states.

Proof sketch

- $\mathcal{R}_c$ can be simulated by adding a padding symbol $\perp$
- New rule: agents in state w can convert others to w

Threshold state complexity: sublogarithmic bounds

Theorem **STACS'18**

There exist protocols $P_0, P_1, \ldots$ and numbers $c_0 < c_1 < \cdots$ such that P_i computes $x \geq c_i$ and has $O(\log \log c_i)$ states.

Proof sketch

- $\mathcal{R}_c$ can be simulated by adding a padding symbol $\perp$
- New rule: agents in state w can convert others to w
- Simulate $\mathcal{R}_c$ from $\{x, y, \perp, \perp, \ldots, \perp\}$

Threshold state complexity: sublogarithmic bounds

Theorem **STACS'18**

There exist protocols $P_0, P_1, \ldots$ and numbers $c_0 < c_1 < \cdots$ such that P_i computes $x \geq c_i$ and has $O(\log \log c_i)$ states.

Proof sketch

- $\mathcal{R}_c$ can be simulated by adding a padding symbol $\perp$
- New rule: agents in state w can convert others to w
- Simulate $\mathcal{R}_c$ from $\{x, y, \perp, \perp, \ldots, \perp\}$
- $\{w, w, \ldots, w\}$ reachable $\iff$ initially $\geq 2^{2^c}$ agents in $\perp$

Threshold state complexity: sublogarithmic bounds

Theorem **STACS'18**

There exist protocols $P_0, P_1, \ldots$ and numbers $c_0 < c_1 < \cdots$ such that P_i computes $x \geq c_i$ and has $O(\log \log c_i)$ states.

Proof sketch

- $\mathcal{R}_c$ can be simulated by adding a padding symbol $\perp$
- New rule: agents in state w can convert others to w
- Simulate $\mathcal{R}_c$ from $\{x, y, \perp, \perp, \ldots, \perp\}$
- $\{w, w, \ldots, w\}$ reachable $\iff$ initially $\geq 2^{2^c}$ agents in $\perp$
- By reversibility and fairness, cannot avoid $\{w, w, \ldots, w\}$

State complexity: beyond threshold

Let $A \in \mathbb{Z}^{m \times k}$, let $\boldsymbol{c} \in \mathbb{Z}^m$ and let n be the largest absolute value of numbers occurring in A and $\boldsymbol{c}$.

Observation

Classical protocol computing $A\boldsymbol{x} + \boldsymbol{c} > \boldsymbol{0}$ has $O(n^m)$ states.

State complexity: beyond threshold

Let $A \in \mathbb{Z}^{m \times k}$, let $\boldsymbol{c} \in \mathbb{Z}^m$ and let n be the largest absolute value of numbers occurring in A and $\boldsymbol{c}$.

Observation

Classical protocol computing $A\boldsymbol{x} + \boldsymbol{c} > \boldsymbol{0}$ has $O(n^m)$ states.

Theorem STACS'18

There exists a protocol that computes $A\boldsymbol{x} + \boldsymbol{c} > \boldsymbol{0}$ and has

- at most $O((m + k) \cdot \log mn)$ states
- at most $O(m \cdot \log mn)$ leaders

Conclusion

Peregrine:

- Graphical and command-line tool for designing, simulating and verifiying population protocols
- Can verify silent protocols

Future work:

- Verification of non silent protocols (ongoing with Amrita)
- Convergence speed analysis (ongoing with Javier and Tony)
- Failure ratio analysis
- LTL model checking

Conclusion

Peregrine:

- Graphical and command-line tool for designing, simulating and verifiying population protocols
- Can verify silent protocols

Future work:

- Verification of non silent protocols (ongoing with Amrita)
- Convergence speed analysis (ongoing with Javier and Tony)
- Failure ratio analysis
- LTL model checking

Conclusion

State complexity:

- Complexity of $x \geq c$ can be decreased from $O(c)$ to $O(\log c)$ and sometimes $O(\log \log c)$
- Similar results for systems of linear inequalities

Future work:

- Is $O(\log \log \log c)$ sometimes possible?

  (not for the class of 1-aware protocols)
- State complexity of Presburger-definable predicates
- Study of the trade-off between size and speed

Conclusion

State complexity:

- Complexity of $x \geq c$ can be decreased from $O(c)$ to $O(\log c)$ and sometimes $O(\log \log c)$
- Similar results for systems of linear inequalities

Future work:

- Is $O(\log \log \log c)$ sometimes possible?

  (not for the class of 1-aware protocols)
- State complexity of Presburger-definable predicates
- Study of the trade-off between size and speed

Thank you!   Vielen Dank!